Journal of Applied Mathematics and Physics, 2020, 8, 2509-2518
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

An Enhanced Steepest Descent Method for Global Optimization-Based Mesh Smoothing

Kang Zhao[1], Yabang Ma[1], You Wang[1], Xin Yin[1], Yufei Guo[2*]

[1]Research and Development Centre, Sichuan Aerospace Chuannan Initiating Explosive Technology Limited, Luzhou, China
[2]Department of Mechanics and Engineering Science, College of Engineering, Peking University, Beijing, China
Email: *yfguo@pku.edu.cn

How to cite this paper: Zhao, K., Ma, Y.B., Wang, Y., Yin, X. and Guo, Y.F. (2020) An Enhanced Steepest Descent Method for Global Optimization-Based Mesh Smoothing. *Journal of Applied Mathematics and Physics*, **8**, 2509-2518.
https://doi.org/10.4236/jamp.2020.811186

Received: May 5, 2020
Accepted: November 23, 2020
Published: November 26, 2020

Abstract

In order to speed up the global optimization-based mesh smoothing, an enhanced steepest descent method is presented in the paper. Numerical experiment results show that the method performs better than the steepest descent method in the global smoothing. We also presented a physically-based interpretation to explain why the method works better than the steepest descent method.

Keywords

Mesh, Mesh Smoothing, Global Mesh Smoothing, Optimization-Based, Steepest Descent Method

1. Introduction

Mesh quality is import for the finite element method. Mesh with high quality can reduce the time to solution, and increase the solution accuracy [1] [2]. Therefore, mesh quality improvement methods are often used in the mesh generation [3] [4]. There are three types of mesh improvement methods: node insertion/deletion optimization [5] [6] [7], topological optimization [8] [9] [10] [11] and geometrical optimization [12]-[23]. Geometrical optimization is also called smoothing, since it improves the mesh by relocating mesh vertices, while preserving mesh topology. For this reason, smoothing has an important role in mesh optimization.

Laplacian smoothing and optimization-based smoothing are two main smoothing methods. Laplacian smoothing method is very efficient, since it calculates the node movements straightforward. But its optimization performance is not as good as optimization-based smoothing, and it may even result in some invalid elements. Some variations of Laplacian smoothing are presented in order to

overcome these disadvantages. Smart Laplacian smoothing is one commonly used variation [14] [24]. Other variations are based on geometric element transformation [16] [17] [19] [25] or angle improvement [15]. Optimization-based smoothing method uses mesh quality to define a cost function, then treats the mesh optimization problem as a cost minimisation problem [26]. Optimization-based smoothing can avoid bad elements and obtain a higher quality mesh. However, its computational cost is higher.

There is a famous optimization-based smoothing Toolkit called Mesquite [27]. Its solver methods include steepest descent method, conjugate gradient method, quasi-Newton method, trust region method, and feasible Newton method [28] [29]. The steepest descent method and the conjugate gradient method are gradient-based, whereas the remaining three are Hessian-based.

Optimization-based smoothing can be divided into global optimization-based smoothing and local optimization-based smoothing. The global smoothing is an all-vertex method where the positions of all free vertices are moved simultaneously within a single iteration, and the local smoothing is a single-vertex method where the position of only one vertex is modified at a time. The performance of the global optimization-based smoothing is better than that of the local optimization-based smoothing [29] [30]. However, the number of variables for global smoothing is proportional to the mesh node number. With the increasing of the mesh nodes, the Hessian-based methods are more unsuitable for global smoothing. Thus, the gradient-based methods appear to be dominant in the global optimization-based smoothing. The conjugate gradient method has been shown to be superior to the steepest descent in most applications [31]. However, the conjugate gradient method requires more storage of intermediate results and more computational cost in a single iteration than the steepest descent method. In addition, the conjugate gradient method is less robust where the cost function surface is relatively flat [32].

Therefore, improving performance of the steepest descent method will facilitate the global optimization-based smoothing. We presented a method which works better than the steepest descent method, but with same storage and almost negligible added computational cost.

The rest of the paper is organized as follows. In Section 2, we briefly review the steepest descent method and point out its shortcomings in mesh optimization. In Section 3, we describe the enhanced steepest descent method and present a physically-based interpretation to explain why the method works better than the steepest descent method. In Section 4, some numerical experiments for triangular meshes and tetrahedral meshes with different initial configurations and different scale problems are performed to compare the steepest descent method and the enhanced method. Finally, we give some conclusions and summarize our study.

2. Steepest Descent Method and Its Drawbacks

The steepest descent method is a line search technique which takes a step along

the gradient direction a teach iteration. Let $f(\mathbf{X})$ be the cost function, where $\mathbf{X}$ is the variable vector representing all variables in the mesh (the coordinates of free nodes). The steepest descent method modifies the vector at t-th iteration according to:

$$\Delta \mathbf{X}_t = -\varepsilon_t \nabla_{\mathbf{X}} f(\mathbf{X}_t) \tag{1}$$

where $\Delta \mathbf{X}_t$ is the change of the variable vector at t-th iteration, $\nabla_{\mathbf{X}}$ is the gradient operator with respect to the variable vector $\mathbf{X}$, and ε_t is a small positive number called step length.

For global optimization-based smoothing, the dimension of the variable vector may be very big. For example, the dimension of the variable vector of a 2D mesh with 10,000 free nodes can be 20,000. The greater the dimension of the variable vector is, the more likely the gradient of the cost function will generate some special circumstances which will affect the convergence of the entire variable vector, such as the values in some dimensions of the gradient of the cost function are always close to 0, but in other dimensions of the gradient of the cost function are always lager. In this circumstance, the steepest descent is particularly slow. We show this case with a mesh in 2D consisting of 4 elements and 1 free node, as shown in **Figure 1**.

As shown in **Figure 2**, the cost function surface of the mesh seems like a narrow valley, the direction of the gradient is almost perpendicular to the long axis of the valley. The free node thus oscillates back and forth in the direction of the short axis, and moves very slowly along the long axis of the valley as shown in **Figure 3**.

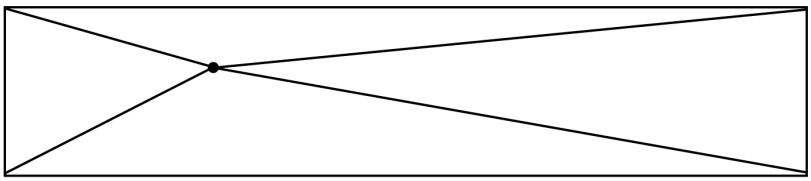


Figure 1. The mesh with 4 elements and 1 free node.

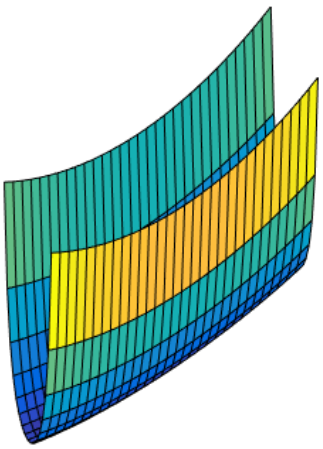


Figure 2. The cost function surface of the mesh.

3. Enhanced Steepest Descent Method

In order to overcome the shortcomings of the steepest descent method, we present a variation of steepest descent method with the inclusion of a special term, as bellow:

$$\Delta \mathbf{X}_t = -\varepsilon_t \nabla_{\mathrm{X}} f\left(\mathbf{X}_t\right) + p\Delta \mathbf{X}_{t-1} \tag{2}$$

where p is the weight factor. The formula considers the change of the variable vector at the current time step with both the current gradient of the cost function and the weight change of the variable vector in previous step.

The rationale for use of the special term is that it can offset fluctuations in the direction of the short axis and speed up the movement in the direction of the long axis as shown in **Figure 4**.

The effect of the special term can also be illustrated with the addition of vectors. The special term tends to expend the current change of variable vector when the direction of current gradient of cost function is similar to that of the previous change, as shown in **Figure 5(a)**, and tends to shorten the current change of variable vector when the direction of current gradient of cost function is different from that of the previous change, as shown in **Figure 5(b)**.

Considering the momentum method:

$$\Delta \mathbf{X}_t = -\varepsilon_t \nabla_{\mathrm{X}} f\left(\mathbf{X}_t\right) + p\Delta \mathbf{X}_{t-1} \tag{3}$$

We can further obtain the relationship between $\Delta \mathbf{X}_t$ and $\Delta \mathbf{X}_{t-n}$ by recursive, as bellow:

$$\Delta \mathbf{X}_t = -\varepsilon_t \nabla_{\mathrm{X}} f\left(\mathbf{X}_t\right) - p\varepsilon_{t-1} \nabla_{\mathrm{X}} f\left(\mathbf{X}_{t-1}\right) + p^2 \Delta \mathbf{X}_{t-2} \tag{4}$$

$$\Delta \mathbf{X}_t = -\varepsilon_t \nabla_{\mathrm{X}} f\left(\mathbf{X}_t\right) - p\varepsilon_{t-1} \nabla_{\mathrm{X}} f\left(\mathbf{X}_{t-1}\right) - p^2 \varepsilon_{t-2} \nabla_{\mathrm{X}} f\left(\mathbf{X}_{t-2}\right) - \cdots + p^n \Delta \mathbf{X}_{t-n} \tag{5}$$

Obviously, the larger p is, the more effect of $\Delta \mathbf{X}_{t-n}$ on $\Delta \mathbf{X}_t$ is. when the directions

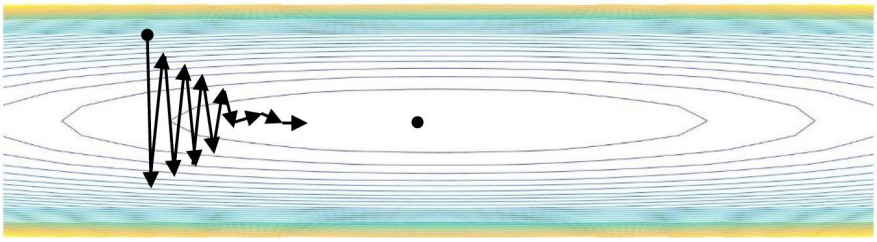


Figure 3. The node movement track of the steepest descent method.

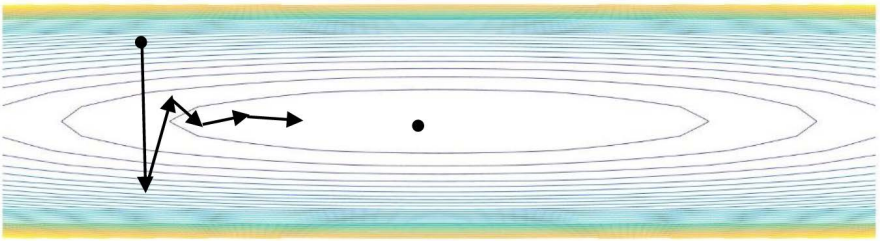


Figure 4. The node movement track of the enhanced method.

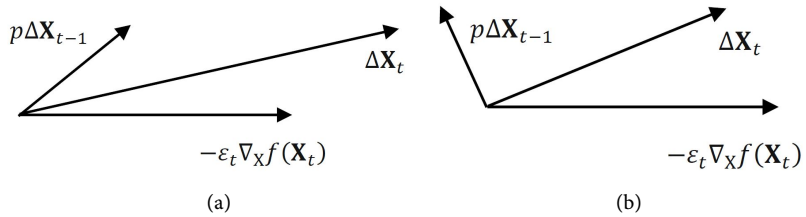


Figure 5. The effect of the momentum term. (a) The role of expending the current change; (b) The role of shortening the current change.

of $\Delta\mathbf{X}_{t-i}$ ($i = 1, \ldots, n$) are similar multiple times in succession, more accumulation of them is beneficial to the variable vector in current step converging to the optimal values at a faster speed. However, a larger p also means that it is difficult to stop when the variable vector reaches the optimal point. With extensive experiments, we found that it is effective to set the weight factor p to be 0.2.

4. Numerical Experiments

In this section, we will report results from a set of numerical experiments designed to compare the steepest descent method and the enhanced method. We consider triangular meshes and tetrahedral meshes separately.

Mesquite has implemented the steepest descent method already. We implement the enhanced method on the basis Mesquite. For step length ε, the Toolkit selects a big value and changes it to satisfy the Armijo condition [33]. When the Armijo condition is not met, the step length is reduced by a reduction factor; and when the step results in a tangled mesh, the step length is reduced by a back-tracking factor. We also employ the default parameter values in the enhanced method, where reduction factor is 0.5 and back-tracking factor is 0.2. Another important consideration in the study is the choice of cost function. We define the cost function as the sum of all element qualities according to the aspect ratio metric.

Aspect ratio quality metric is a commonly used quality metric in mesh smoothing. The quality value of an element with different configurations can vary significantly according to the metric. Various formulas have been used to compute the aspect ratio quality metric [3] [28]. We use a simple one implemented in Mesquite. For triangular element, it is defined as:

$$(l_1^2 + l_2^2 + l_3^2)/(4\sqrt{3}\times S) \tag{6}$$

And for tetrahedral element, it is defined as:

$$(l_1^2 + l_2^2 + l_3^2 + l_4^2 + l_5^2 + l_6^2)/(36\sqrt{2}\times V) \tag{7}$$

where l_i represents edge length, S represents area of triangular element, and V represents volume of tetrahedral element. The value of this metric range from 1 to ∞. The optimal value of the metric is 1.

Since the objective function used in our numerical experiments is non-convex, the different methods may converge to different local minimums. To ensure that

this does not affect our experiments, we verified whether all methods converged to the same optimal mesh for each experiment by comparing the vertex coordinates of the optimal meshes. Unless otherwise stated, the methods converged to the same optimal mesh. In all experiments, the solution is considered optimal when it has converged to six significant digits.

4.1. Triangular Mesh Experiments

The triangular mesh examples are four Delaunay triangulations with different number of random points in the unit square as shown in **Figures 6(a)-(d)**. Their uniformly distributed boundary nodes are fixed during the optimization process. The related results are listed in **Table 1**. It can be seen that the enhanced steepest descent method is far more efficient than the steepest descent method.

4.2. Tetrahedral Mesh Experiments

We chose three tetrahedral meshes with big scale problems and different initial configurations from mesquite example files as shown in **Figures 7(a)-(c)**. Their boundary nodes are also fixed during the optimization process. The related results are listed in **Table 2**. It can also be seen that the enhanced steepest descent

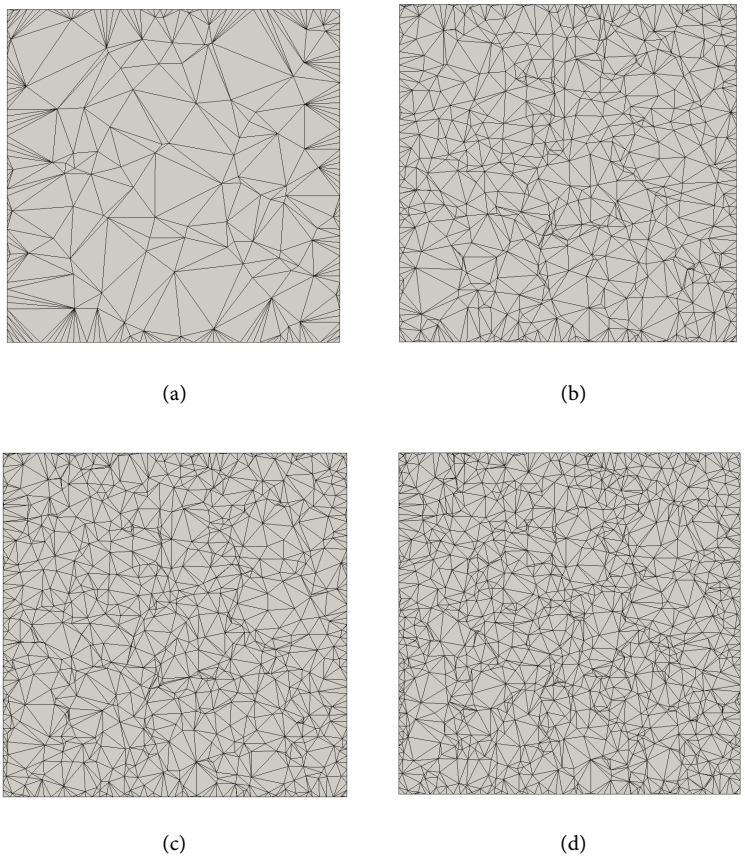
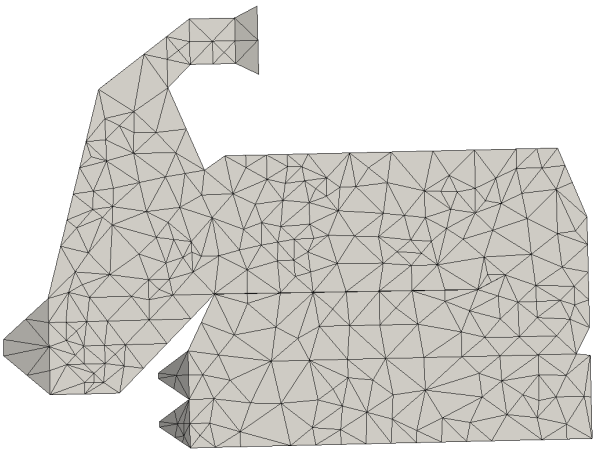


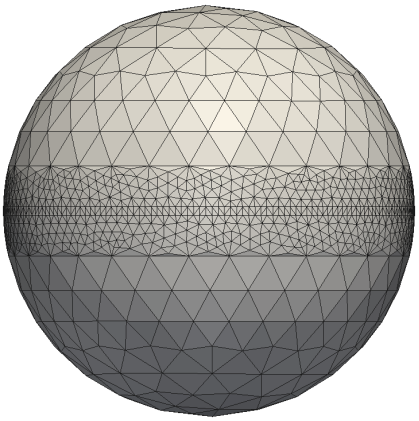
Figure 6. Four triangular meshes. (a) The first mesh with 398 elements; (b) The second mesh with 1198 elements; (c) The third mesh with 1598 elements; (d)The fourth mesh with 2198 elements.

Table 1. The performance of the two methods for triangular meshes.

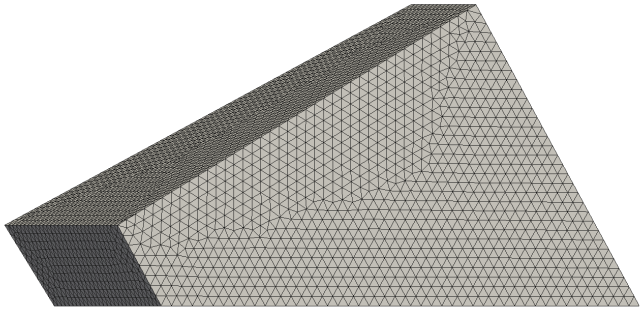
Meshes	Free nodes	Elements	Methods	Cost function	Time/ (s)
Case 1	100	398	Initial	894.763	0.000
			Steepest descent	649.543	0.009
			Ours	649.543	0.005
Case 2	500	1198	Initial	2211.954	0.000
			Steepest descent	1433.80	0.027
			Ours	1433.80	0.019
Case 3	700	1598	Initial	3030.91	0.000
			Steepest descent	1859.26	0.036
			Ours	1859.26	0.028
Case 4	1000	2198	Initial	4101.56	0.000
			Steepest descent	2538.92	0.048
			Ours	2538.92	0.034



(a)



(b)



(c)

Figure 7. Three tetrahedral meshes. (a) The first mesh with 112393 elements; (b) The second mesh with 23636 elements; (c) The third mesh with 11098 elements.

Table 2. The performance of the two methods for tetrahedral meshes.

Meshes	Free nodes	Elements	Methods	Cost function	Time/(s)
Case 1	2750	11,098	Initial	14,174.3	0
			Steepest descent	13,673.2	0.186
			Ours	13,673.2	0.122
Case 2	4793	23,636	Initial	30,223.5	0
			Steepest descent	28,322.6	0.368
			Ours	28,322.6	0.275
Case 3	21156	112,393	Initial	1,307,300	0
			Steepest descent	1,252,959	1.813
			Ours	1,252,959	1.251

method is more efficient than the steepest descent method.

5. Conclusions

In this paper, we presented an enhanced steepest descent method with same storage and almost negligible added computational cost as gradient descent method. Numerical experiments show that it can improve the speed of the mesh smoothing.

We presented a physically-based interpretation of the enhanced method. This give us insights explaining why it can work better than steepest descent method in global mesh smoothing.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Kim, J., Sastry, S.P. and Shontz, S.M. (2012) A Numerical Investigation on the Interplay amongst Geometry, Meshes, and Linear Algebra in the Finite Element Solu-

tion of Elliptic Pdes. *Engineering with Computers,* **28**, 431-450. https://doi.org/10.1007/s00366-011-0231-0

[2] Freitag, L. and Ollivier-Gooch, C. (2000) A Cost/Benefit Analysis of Simplicial Mesh Improvement Techniques as Measured by Solution Efficiency. *Int. J. Comput. Geometry Appl.,* **10**, 361-382. https://doi.org/10.1142/S0218195900000218

[3] Guo, Y.F., Hai, Y.Q. and Liu, J.F. (2020) Direct Modifications of Tetrahedral Meshes. *Engineering Computations*. https://doi.org/10.1108/EC-12-2019-0573

[4] Guo, Y.F., Shang, F.F. and Liu, J.F. (2018) Surface Adaptive Mesh Generation for STL Models Based on Ball-Packing Method. *Journal of Computer-Aided Design & Computer Graphics*, **30**, 549-556. https://doi.org/10.3724/SP.J.1089.2018.16515

[5] Bank, R.E., Sherman, A.H. and Weiser, A. (1983) Chapter Refinement Algorithms and Data Structures for Regular Local Mesh Refinement. *Journal of Scientific Computing*.

[6] Turk, G. (2001) Re-Tiling Polygonal Surfaces. ACM SIGGRAPH Computer Graphics. 26. https://doi.org/10.1145/142920.134008

[7] María-Cecilia, R. (1997) New Longest-Edge Algorithms for the Refinement and/or Improvement of Unstructured Triangulations. *International Journal for Numerical Methods in Engineering,* **40**, 3313-3324. https://doi.org/10.1002/(SICI)1097-0207(19970930)40:18<3313::AID-NME214>3.0.CO;2-#

[8] Chen, X., Peng, D. and Gao, S. (2013) Svm-Based Topological Optimization of Tetrahedral Meshes. https://doi.org/10.1007/978-3-642-33573-0_13

[9] Chen, J., Zheng, J., Zheng, Y., Si, H., Hassan, O. and Morgan, K. (2017) Improved Boundary Constrained Tetrahedral Mesh Generation by Shell Transformation. *Applied Mathematical Modelling,* **51**, 764-790. https://doi.org/10.1016/j.apm.2017.07.011

[10] Liu, J.F., Sun, S.L. and Chen, Y.Q. (2012) A New Method of Quality Improvement for Quadrilateral Mesh Based on Small Polygon Reconnection. *Acta Mechanica Sinica,* **28**, 140-145. https://doi.org/10.1007/s10409-012-0022-x

[11] George, P. and Borouchaki, H. (2003) Back to Edge Flips in 3 Dimensions. 393-402.

[12] Field, D.A. (1988) Laplacian Smoothing and Delaunay Triangulations. *Communications in Numerical Methods in Engineering,* **4**, 709-712. https://doi.org/10.1002/cnm.1630040603

[13] Lo, S.H. (1985) A New Mesh Generation Scheme for Arbitrary Planar Domains. *International Journal for Numerical Methods in Engineering,* **21**, 1403-1426. https://doi.org/10.1002/nme.1620210805

[14] Vollmer, J., Mencl, R. and H.MÃ¼ller (1999) Improved Laplacian Smoothing of Noisy Surface Meshes. John Wiley and Sons, Ltd., Vol. 18, 131-138. https://doi.org/10.1111/1467-8659.00334

[15] Zhou, T. and Shimada, K. (2000) An Angle-Based Approach to Two-Dimensional Mesh Smoothing. *The 9th International Meshing Roundtable*.

[16] Vartziotis, D., Athanasiadis, T., Goudas, I. and Wipper, J. (2008) Mesh Smoothing Using the Geometric Element Transformation Method. *Computer Methods in Applied Mechanics and Engineering,* **197**, 3760-3767. https://doi.org/10.1016/j.cma.2008.02.028

[17] Vartziotis, D. and Wipper, J. (2009) The Geometric Element Transformation Method for Mixed Mesh Smoothing. *Engineering with Computers,* **25**, 287-301. https://doi.org/10.1007/s00366-009-0125-6

[18] Vartziotis, D. and Wipper, J. (2012) Fast Smoothing of Mixed Volume Meshes Based on the Effective Geometric Element Transformation Method. *Computer Methods in Applied Mechanics & Engineering,* **201-204**, 65-81. https://doi.org/10.1016/j.cma.2011.09.008

[19] Sun, S., Zhang, M. and Gou, Z. (2015) Smoothing Algorithm for Planar and Surface Mesh Based on Element Geometric Deformation. *Mathematical Problems in Engineering,* **2015**, 435648.1-435648.9. https://doi.org/10.1155/2015/435648

[20] Lin, T.J., Guan, Z.Q. and Chang, J.H. (2013) An Efficient Method for Unstructured Dynamic Mesh Deformation-Vertex-Ballspring Smoothing. *Journal of Computer-Aided Design &Computer Graphics*, **25**, 1651-1657.

[21] Xu, K., Gao, X. and Chen, G. (2018) Hexahedral Mesh Quality Improvement via Edge-Angle Optimization. *Computers & Graphics*, **70**, 17-27. https://doi.org/10.1016/j.cag.2017.07.002

[22] Xu, H. and Newman, T.S. (2006) An Angle-Based Optimization Approach for 2d Finite Element Mesh Smoothing. *Finite Elements in Analysis & Design,* **42**, 1150-1164. https://doi.org/10.1016/j.finel.2006.01.016

[23] Lo, S.H. (1997) Optimization of Tetrahedral Meshes Based on Element Shape Measures. *Computers & Structures,* **63**, 951-961. https://doi.org/10.1016/S0045-7949(96)00399-9

[24] Chen, Z.J., Tristano, J. and Kwok, W. (2003) Combined Laplacian and Optimization-Based Smoothing for Quadratic Mixed Surface Meshes.

[25] Ahmed, A.G.M., Guo, J., Yan, D.M., Franceschia, J.Y., Zhang, X. and Deussen, O. (2017) A Simple Push-Pull Algorithm for Blue-Noise Sampling. *IEEE Transactions on Visualization & Computer Graphics,* **23**, 2496-2508. https://doi.org/10.1109/TVCG.2016.2641963

[26] Leordeanu, M. and Hebert, M. (2008) Smoothing-Based Optimization. 26*th IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), 1-8. https://doi.org/10.1109/CVPR.2008.4587482

[27] Brewer, M., Diachin, L., Knupp, P., Leurent, T. and Melander, D. (2003) The Mesquite Mesh Quality Improvement Toolkit. *Proceedings of the* 12*th International Meshing Roundtable*.

[28] Munson, T. (2007) Mesh Shape-Quality Optimization Using the Inverse Mean-Ratio Metric. **110**, 561-590. https://doi.org/10.1007/s10107-006-0014-3

[29] Diachin, L.F., Knupp, P., Munson, T. and Shontz, S. (2006) A Comparison of Two Optimization Methods for Mesh Quality Improvement. *Engineering with Computers,* **22**, 61-74. https://doi.org/10.1007/s00366-006-0015-0

[30] Sastry, S.P. and Shontz, S.M. (2012) Performance Characterization of Nonlinear Optimization Methods for Mesh Quality Improvement. *Engineering with Computers,* **28**, 269-286. https://doi.org/10.1007/s00366-011-0227-9

[31] Sejnowski, T.J. (2016) The Computational Brain. *Quarterly Review of Biology,* **103**, 574-574.

[32] Qian, N. (1999) On the Momentum Term in Gradient Descent Learning Algorithms. *Neural Networks,* **12**, 145-151. https://doi.org/10.1016/S0893-6080(98)00116-6

[33] Shi, F. (2000) Minimization of Functions Having Lipschitz Continuous Gateaux-Derivatives. *Journal of Mathematics,* **20**, 359-360.